Simulasi Algoritma Jaringan Syaraf Tiruan Order-Satu

Agus Basukesti
Jurusan Teknik Elektro STT Adisutjipto Yogyakarta
Jl. Janti Blok-R Lanud-Adisutjipto

Abstract
Artificial neural network is a computational algorithm that mimics the workings of nerve cells. All the incoming signal is multiplied by a weight that is on each input, by neuronal cells, all signals have been multiplied by weights are summed and then added again with bias. The sum is input to a function (activation function) produces the output of neurons (here used a linear activation function). During the learning process, the weights and biases are always updated using learning algorithms. if there is an error in the output. For the identification process, the weights are directly weighing input is what is called as a search parameter, the price of w1, w2, w3 and w4. In the on-line identification, neurons or networks of neurons will always be 'learned' every input and output data.

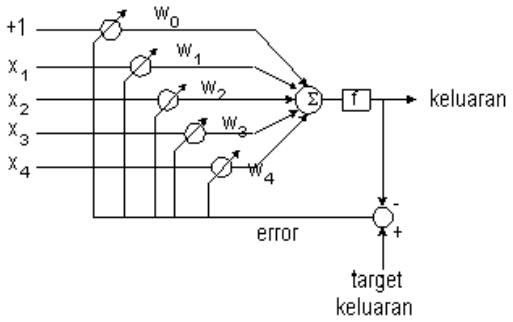
Keywords: neuro, activation function, learning constants.

I. Pendahuluan

Jaringan neural artifial adalah sistem pemrosesan informasi yang mempunyai karakteristik kinerja tertentu seperti jaringan neural biologis. Jaringan neural artificial telah dikembangkan sebagai generalisasi model matematik dari kognisi manusia atau biologi neural. Jaringan syaraf tiruan merupakan algoritma komputasi yang meniru cara kerja sel syaraf. Semua sinyal yang masuk dikalikan dengan bobot yang ada pada tiap masukan, oleh sel neuron, semua sinyal yang sudak dikalikan dengan bobot dijumlahkan kemudian ditambah lagi dengan bias. Hasil penjumlahan ini diinputkan ke suatu fungsi (fungsi aktifasi) menghasilkan keluaran dari neuron (di sini digunakan fungsi aktifasi linier). Selama proses pembelajaran, bobot-bobot dan bias selalu diperbaharui menggunakan algoritma belajar. jika ada error pada keluaran.

II. Metodologi

Dalam proses identifikasi, bobot-bobot yang yang secara langsung memboboti masukan inilah yang dinamakan sebagai parameter yang dicari, seperti terlihat pada Gambar 1, parameter yang dicari adalah harga w_1, w_2, w_3 dan w_4. Dalam identifikasi secara on-line, neuron ataupun jaringan neuron akan selalu 'belajar' setiap ada data masukan dan keluaran.



Gambar 1. Sel neuron ketika sedang melakukan proses belajar

Algoritma untuk memperbaharui bobot pada neuron satu lapis adalah seperti pada bagian algoritma pemrograman Jaringan Syaraf Tiruan satu lapis langkah ke-7. Sedangkan untuk Jaringan Syaraf Tiruan dua lapis adalah seperti pada bagian algoritma pemrograman JST dua lapis langkah ke-8 dan 9.

III. Teori

3.1 Logika Fuzzy

Logika *fuzzy* yang pertama kali diperkenalkan oleh Lotfi A. Zadeh, memiliki derajat keanggotaan dalam rentang 0(nol) hingga 1(satu), berbeda dengan logika digital yang hanya memiliki dua nilai yaitu 1(satu) atau 0(nol). Logika *fuzzy* digunakan untuk menerjemahkan suatu besaran yang diekspresikan menggunakan bahasa (*linguistic*), misalkan besaran kecepatan laju kendaraan yang diekspresikan dengan pelan, agak cepat,

cepat dan sangat cepat. Secara umum dalam sistem logika *fuzzy* terdapat empat buah elemen dasar, yaitu:

1. Basis kaidah (*rule base*), yang berisi aturan-aturan secara linguistik yang bersumber dari para pakar;

2. Suatu mekanisme pengambilan keputusan (*inference engine*), yang memperagakan bagaimana para pakar mengambil suatu keputusan dengan menerapkan pengetahuan (*knowledge*);

3. Proses fuzzifikasi (*fuzzification*), yang mengubah besaran tegas (*crisp*) ke besaran *fuzzy*;

4. Proses defuzzifikasi (*defuzzification*), yang mengubah besaran *fuzzy* hasil dari *inference engine*, menjadi besaran tegas (*crisp*).

Fuzzy Membership, jika *X* adalah suatu kumpulan obyek-obyek dan *x* adalah elemen dari *X*. Maka himpunan *fuzzy A* yang memiliki domain *X* didefinisikan sebagai:

$$A = \{(x, \mu_A(x)) \mid x \in X \}$$

dimana nilai $\mu_A(x)$ berada dalam rentang 0 hingga 1.

Fuzzy Membership Operation, Seperti pada himpunan klasik, himpunan *fuzzy* juga memiliki operasi himpunan yang sama yaitu gabungan (*union*), irisan (*intersection*) dan komplemen. **Union (Gabungan),** gabungan dari dua buah himpunan *fuzzy A* dan *B* adalah himpunan *fuzzy C* ditulis sebagai $C = A \cup B$ atau $C = A OR B$, memiliki fungsi keanggotaan yang berhubungan dengan *A* dan *B* yang didefinisikan sebagai berikut:

$$\mu_C(x) = \max(\mu_A(x), \mu_B(x)) = \mu_A(x) \vee \mu_B(x)$$

$$\mu_C(x) = S(\mu_A(x), \mu_B(x)) = \mu_A(x) \tilde{+} \mu_B(x)$$

dengan $\tilde{+}$ adalah operator biner untuk fungsi *S* dan biasa disebut sebagai operator *T-conorm* atau *S-norm*, yang memiliki sifat-sifat sebagai berikut:

S(1,1) = 1, S(0,a)=S(a,0)= a	(*boundary*);
S(a,b) $\leq$ *S(c,d) jika a* $\leq$ *c dan b* $\leq$ *d*	(*monotonicity*);
S(a,b)=S(b,a)	(*commutativity*);
S(a,S(b,c)) = S(S(a,b),c)	(*associativity*).

Intersection (Irisan), irisan dari dua buah himpunan *fuzzy A* dan *B* adalah himpunan *fuzzy C* dituliskan sebagai $C = A \cap B$ atau $C = A AND B$, memiliki fungsi keanggotaan yang berhubungan dengan *A* dan *B* yang didefinisikan sebagai berikut:

$$\mu_C(x) = \min(\mu_A(x), \mu_B(x)) = \mu_A(x) \wedge \mu_B(x)$$

$$\mu_C(x) = T(\mu_A(x), \mu_B(x)) = \mu_A(x) \tilde{*} \mu_B(x)$$

dengan $\tilde{*}$ adalah operator bineri untuk fungsi *T*, yang biasa disebut sebagai operator *T-norm*, yang memiliki sifat-sifat sebagai berikut:

T(0,0) = 0, T(a,1) = T(1,a) = a	(***boundary***);
T(a,b) $\leq$ ***T(c,d) jika a*** $\leq$ ***c dan b*** $\leq$ ***d***	(***monotonicity***)
T(a,b)= T(b,a)	(***commutativity***)
T(a,T(b,c)) = T(T(a,b),c)	(***associativity***).

Fuzzy Set Membership Function, fungsi-fungsi keanggotaan *fuzzy* terparameterisasi satu dimensi yang umum digunakan diantaranya adalah:

1. Fungsi keanggotaan segitiga, disifati oleh parameter{*a,b,c*} yang didefinisikan sebagai berikut:

$$segitiga(x; a, b, c) = \begin{cases} 0, & x \leq a \\ \dfrac{x-a}{b-a}, & a \leq x \leq b \\ \dfrac{c-x}{c-b}, & b \leq x \leq c \\ 0, & c \leq x \end{cases}$$

bentuk yang lain dari persamaan di atas adalah

$$segitiga(x; a, b, c) = \max\left(\min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b} \right), 0 \right)$$

parameter {*a*,*b*,*c*} (dengan *a*<*b*<*c*) yang menentukan koordinat *x* dari ketiga sudut segitiga tersebut, seperti terlihat pada Gambar 3(a).

2. Fungsi keanggotaan trapesium, disifati oleh parameter{*a*,*b*,*c*,*d*} yang

didefinisikan sebagai berikut:

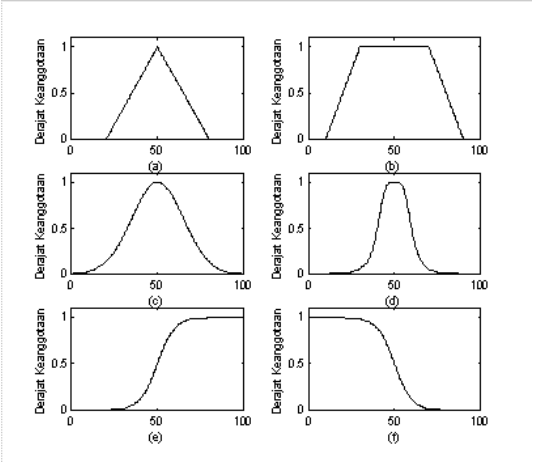
$$trapesium(x;a,b,c,d) = \begin{cases} 0, & x \le a \\ \frac{x-a}{b-a}, & a \le x \le b \\ 1, & b \le x \le c \\ \frac{d-x}{d-c}, & c \le x \le d \\ 0, & d \le x \end{cases}$$

parameter {*a*,*b*,*c*,*d*} (dengan *a*<*b*<*c*<*d*) yang menentukan koordinat *x* dari keempat sudut trapesium tersebut, seperti terlihat pada Gambar 3(b).

3. Fungsi keanggotaan Gaussian, difati oleh parameter {*c*,*s*} yang didefinisikan sebagai berikut:

$$gaussian\ (x;c,\sigma) = e^{-\frac{1}{2}(\frac{x-c}{\sigma})^2}$$

Fungsi keanggotaan Gauss ditentukan oleh parameter *c* dan *s* yang menunjukan titik tengah dan lebar fungsi, seperti terlihat pada Gambar 3(c) .



Gambar 3. Kurva fungsi keanggotaan, (a).segitiga(*x*;20,50.80), (b).trapesium (*x*;10,30,70,90), (c).gaussian(*x*;50,15), (d).bell(*x*;10,2,50), (e).sigmoid (*x*;0.2,50) dan (f).sigmoid(*x*;-0.2,50).

4. Fungsi keanggotaan *generalized bell*, disifati oleh parameter {*a*,*b*,*c*} yang didefinisikan sebagai berikut:

$$bell(x;a,b,c) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}$$

parameter *b* selalu positif,

supaya kurva menghadap kebawah, seperti terlihat pada Gambar 3(d).

5. Fungsi keanggotaan sigmoid, disifati oleh parameter {*a*,*c*} yang didefinisikan sebagai berikut:

$$sig(x;a,c) = \frac{1}{1+\exp[-a(x-c)]}$$

parameter *a* digunakan untuk menentukan kemiringan kurva pada saat *x*=*c*. Polaritas dari *a* akan menentukan kurva itu kanan atau kiri terbuka, seperti terlihat pada Gambar 3.(d) dan 3.(e).

Sebelumnya akan didefinisikan dulu mengenai himpunan bagian yang memiliki peranan penting dalam himpunan *fuzzy*.

Fuzzy IF-Then Rule, kaidah *fuzzy If-Then* (dikenal juga sebagai kaidah *fuzzy*, implikasi *fuzzy* atau pernyataan kondisi (*fuzzy*) diasumsikan berbentuk: Jika *x* adalah *A* maka *y* adalah B

Dengan *A* dan *B* adalah nilai linguistik yang dinyatakan dengan himpunan *fuzzy* dalam semesta pembicaraan *X* dan *Y*. Sering kali "*x* adalah *A*" disebut sebagai *antecedent* atau *premise*, sedangkan "*y* adalah *B*" disebut *consequence* atau *conclusion*.

Fuzzy Reasoning, kaidah dasar dalam menarik kesimpulan dari dua nilai logika tradisional adalah *modus ponens*, yaitu kesimpulan tentang nilai kebenaran pada *B* diambil berdasarkan kebenaran pada *A*. Sebagai contoh, jika *A* diidentifikasi dengan "tomat itu merah" dan *B* dengan "tomat itu masak", kemudian jika benar kalau "tomat itu merah" maka "tomat itu masak", juga benar. Konsep ini digambarkan sebagai berikut:

premise 1 (kenyataan)	:	*x* adalah *A*,
premise 2 (kaidah)	:	jika *x* adalah *A* maka *y* adalah *B*.
Consequence (kesimpulan)	:	*y* adalah *B*.

Secara umum dalam melakukan penalaran, *modus ponens* digunakan dengan cara pendekatan. Sebagai contoh, jika ditemukan suatu kaidah implikasi yang sama dengan "jika tomat itu merah maka tomat itu masak", misalnya "tomat itu kurang lebih merah," maka dapat disimpulkan "tomat itu kurang lebih masak", hal ini dapat dituliskan seperti berikut:

premise 1 (kenyataan)	:	*x* adalah *A*',
premise 2 (kaidah)	:	jika *x* adalah *A* maka *y* adalah *B*.
Consequence (kesimpulan)	:	*y* adalah *B*'.

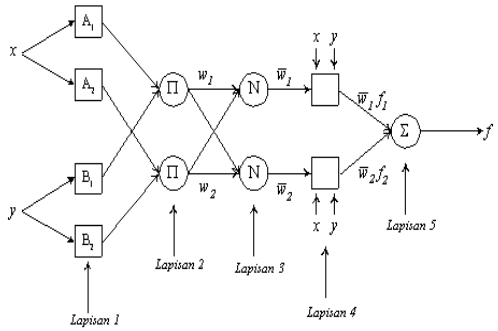
Dengan *A*' adalah dekat ke *A* dan *B*' adalah dekat ke *B*. Ketika *A*, *B*, *A*' dan *B*'adalah himpunan fuzzy dari semesta yang berhubungan, maka penarikan kesimpulan seperti tersebut dinamakan penalaran dengan pendekatan (*approximate reasoning*) yang disebut juga dengan *generalized modus ponens* (*GMP*).

3.2 Adaptive Neuro Fuzzy

Neurofuzzy adalah gabungan dari dua sistem yaitu sistem logika *fuzzy* dan jaringan syaraf tiruan. Sistem *neurofuzzy* berdasar pada sistem inferensi *fuzzy* yang dilatih menggunakan algoritma pembelajaran yang diturunkan dari sistem jaringan syaraf tiruan. Dengan demikian, sistem *neurofuzzy* memiliki semua kelebihan yang dimiliki oleh sistem inferensi fuzzy dan sistem jaringan syaraf tiruan. Dari kemampuannya untuk belajar maka sistem *neurofuzzy* sering disebut sebagai ANFIS (*adaptive neuro fuzzy inference systems*)

3.3 Struktur ANFIS

Salah satu bentuk struktur yang sudah sangat dikenal adalah seperti terlihat pada Gambar 4. Dalam struktur ini, sistem inferensi *fuzzy* yang diterapkan adalah inferensi fuzzy model Takagi-Sugeno-Kang.



Gambar 4. Struktur ANFIS

Seperti terlihat pada Gambar 4, sistem ANFIS terdiri dari 5 lapisan, lapisan yang disimbolkan dengan kotak adalah lapisan yang bersifat adaptif. Sedangkan yang disimbolkan dengan lingkaran adalah bersifat tetap. Setiap keluaran dari masing-masing lapisan disimbolkan dengan $O_{l,i}$ dengan i adalah urutan simpul dan l adalah menunjukan urutan lapisannya. Berikut ini adalah penjelasan untuk setiap lapisan, yaitu:

a. Lapisan 1.

Berfungsi untuk membangkitkan derajat keanggotaan

$$O_{1,i} = \mu_{A_i}(x) \qquad \mathrm{i} = 1,2$$

dan

$$O_{1,i} = \mu_{B_i}(y) \qquad \mathrm{i} = 1,2$$

dengan x dan y adalah masukan bagi simpul ke-i

$$\mu_{A_i}(x) = \frac{1}{1 + \left| \frac{x - c_i}{a_i} \right|^{2b_i}}$$

dengan {a_i, b_i dan c_i}adalah parameter dari fungsi keanggotaan atau disebut sebagai parameter *premise*.

b. Lapisan 2
Berfungsi untuk membangkitkan *firing-strength* dengan mengalikan setiap sinyal masukan.

$$O_{2,1} = w_i = \mu_{A_i}(x) \text{ x } \mu_{B_i}(y) \quad \text{i} = 1,2$$

c. Lapisan 3
Menormalkan *firing strength*

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2} \quad \text{i} = 1,2$$

d. Lapisan 4
Menghitung keluaran kaidah berdasarkan parameter *consequent* {p_i, q_i dan r_i}

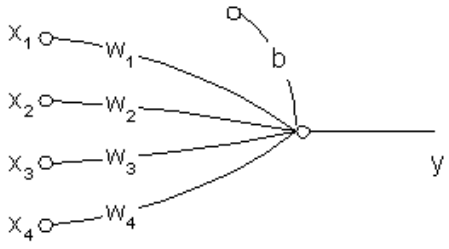
$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_i x + q_i y + r_i)$$

e. Lapisan 5
Menghitung sinyal keluaran ANFIS dengan menjumlahkan semua sinyal yang masuk

$$O_{5,i} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f}{\sum_i w_i}$$

Algoritma Pembelajaran, proses adaptasi yang terjadi dalam sistem ANFIS dikenal juga dengan pembelajaran. Parameter-parameter ANFIS (baik premise maupun consequent) Selama proses belajar akan diperbaharui menggunakan metode pembelajaran. Metode pembelajaran yang digunakan dalam sistem ANFIS adalah algoritma pembelajaran hibrid. Algoritma ini terdiri dari dua bagian yaitu bagian arah maju dan bagian arah mundur. Pada bagian arah maju, proses adaptasi dilakukan menggunakan metode LSE dan terjadi pada parameter consequent. Sedangkan pada bagian arah mundur, proses adaptasi dilakukan menggunakan metode gradient-descent dan terjadi pada parameter premise.

Algorithm Pemrograman JST
a. Untuk Neuron Satu Lapis



Gambar 5. Neuron satu lapis

Algoritma pemrograman untuk neuron satu lapis didasarkan pada Gambar 2, dimana **fungsi aktifasinya linier f(x) = x**, data masukan dinyatakan dengan matrik berikut:

$$\text{X} = \begin{bmatrix} \text{x}_1 & \text{x}_2 & \text{x}_3 & \text{x}_4 \end{bmatrix}$$

bobot-bobot link neuron adalah:

$$\text{W} = \begin{bmatrix} \text{w}_1 & \text{w}_2 & \text{w}_3 & \text{w}_4 \end{bmatrix}$$

bias = b
maka $y = X*W^T + b$, atau $y = x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4 + b$, dengan demikian parameternya adalah q = W.
Algoritma pemrogramannya adalah:

1. Inisialisasi bobot-bobot (termasuk juga bias), termasuk perubahan bobot awal.

2. Mengambil nilai x_1, x_2, x_3 dan x_4 juga nilai target.

3. Menghitung keluaran jaringan neuron
$y = X * W^T + b$

4. Menghitung parameter
$\theta = W$

5. Menghitung error keluaran
e = target – y

6. Menyimpan bobot-bobot ke dalam variabel bobot lama

7. Menghitung perubahan bobot-bobot pada lapisan keluaran

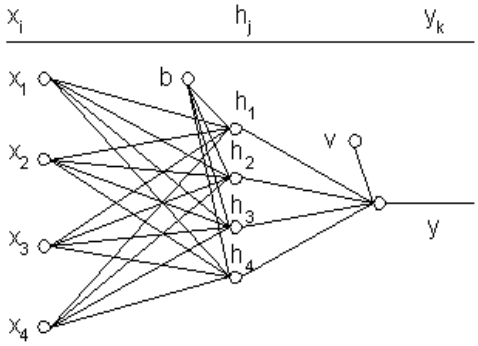
$$\Delta w_i = \eta e x_i \quad \Delta b = \eta e \quad w_i(\text{baru}) = w_i(\text{lama}) + \Delta w_i(\text{baru}) + \alpha \Delta w_i(\text{lama})$$

$$b(\text{baru}) = b(\text{lama}) + \Delta b(\text{baru}) + \alpha \Delta b(\text{lama})$$

8. Menyimpan perubahan-perubahan bobot dan bias ke variabel perubahan lama.

9. Kembali ke langkah 2.

b. Untuk Neuron Dua Lapis



Gambar 6. Neuron dua lapis

Algoritma pemrograman untuk neuron dua lapis didasarkan pada Gambar 3, dimana fungsi aktifasinya linier f(x) = x, data masukan dinyatakan dengan matriks:

$$X = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}$$

Bobot-bobot link neuron masukan adalah a_{ij}, sehingga dalam bentuk matrik menjadi:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

Keluaran dari tiap-tiap neuron pada lapisan masukan adalah:

$$H = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \end{bmatrix}$$

Bobot-bobot bias pada lapisan masukan yaitu:

$$B = \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \end{bmatrix}$$

Bobot-bobot link neuron pada lapisan keluaran yaitu:

$$W = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 \end{bmatrix}$$

bobot bias pada lapisan keluaran = v, keluaran NN adalah, $y = (X*A+B)*W^T + v$.

Algoritma pemrogramannya adalah:

1. Inisialisasi bobot-bobot (termasuk juga bias), termasuk perubahan-perubahan bobot awal.

2. Mengambil nilai x_1, x_2, x_3 dan x_4 juga nilai target.

3. Menghitung keluaran jaringan neuron

   $y = (X * A + B) * W^T + v$

4. Menghitung parameter

   $\theta = A * W^T$

5. Menghitung error keluaran

   e = target – y

6. Menyimpan bobot-bobot ke dalam variabel bobot lama

7. Menghitung matrik H

   H = X*A + b

8. Menghitung error propagasi pada lapisan keluaran

   $\delta_k = e * f'(y\_in_k)$

Menghitung perubahan bobot-bobot pada lapisan keluaran

$$\Delta w'_{jk} = \eta \delta_k h_j \qquad \Delta v_k = \eta \delta_k$$

$$w_{jk}(\text{baru}) = w_{jk}(\text{lama}) + \Delta w_{jk}(\text{baru}) + \alpha \Delta w_{jk}(\text{lama})$$

$$v_k(\text{baru}) = v_k(\text{lama}) + \Delta v_k(\text{baru}) + \alpha \Delta v_k(\text{lama})$$

9. Menghitung error propagasi pada lapisan masukan :

$$\delta_j = f'(hc\_in_j) \sum_{1}^{j} \delta_k w_{jk}$$

$$\Delta a_{ij} = \eta \delta_j x_i \qquad \Delta b_j = \eta \delta_j$$

$$a_{ij}(\text{baru}) = a_{ij}(\text{lama}) + \Delta a_{ij}(\text{baru}) + \alpha \Delta a_{ij}(\text{lama})$$

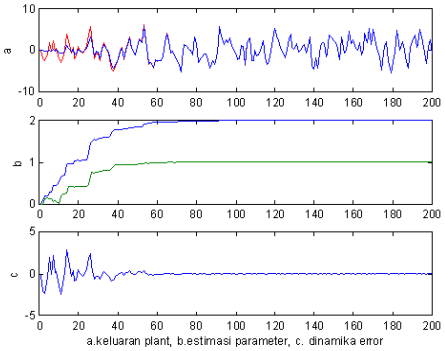
$$b_j(\text{baru}) = b_j(\text{lama}) + \Delta b_j(\text{baru}) + \alpha \Delta b_j(\text{lama})$$

10. Menyimpan perubahan bobot-bobot dan bias kedalam variabel perubahan lama.

11. Kembali ke langkah 2.

IV. Hasil Simulasi

Plant ke-1 yaitu y(k) = 2u(k) + u(k-1), konstanta pembelajaran (delta rule bias) b=0,05; a=0, bobot awal 0, $q_1 = 2$ dan $q_2 = 1$



Gambar 7. Hasil simulasi identifikasi sistem orde 1 plant ke-1

Hasil keluaran plant gambar 7a, nampak pada nilai sinyal 60 keluaran tidak ada *error* (galat). Estimasi parameter pada nilai sinyal 80 nampak sangat jelas (gambar 7b) dan pejalanan *error* berfluktuasi dari sinyal 0-60, pada posisi nol pada nilai sinyal di atas 60

V. Kesimpulan

Dari pembahasan di atas dapat disimpulkan sebagai berikut:

a. Algoritma pemrograman Jaringan Syaraf Tiruan untuk neuron digunakan **fungsi aktifasinya linier f(x) = x**, data masukan dinyatakan dengan matrik X bobot W, bias = b maka y = $X*W^T$ + b, atau y = $x_1w_1 + x_2w_2 + x_3w_3 + x_4w_4$ + b, dengan demikian parameternya adalah q= W.

b. Dari simulasi Jaringan Syaraf Tiruan plant-1 Gambar 4 dengan parameter konstanta bembelajaran b=0,05; a=0, bobot awal 0, $q_1 = 2$ dan $q_2 = 1$, bahwa nilai keluaran plant, estimasi parameter, dinamika *error* sinyal pada nilai 60.

Daftar Pustaka

[1] Jang, J.S.R.., Sun, C.T., E. Mizutani., *Neuro-Fuzzy and Soft Computing,* Prentice-Hall, New Jersey,1997.

[2] Sri Widodo Th., *Sistem Neuro Fuzzy*, Graha Ilmu, Yogyakarta,2005

[3] Fausett L., *Fundamentals of Neural Networks*, Prentice-Hall, New Jersey,1994.

[4] Kosko B., *Neural Networks and Fuzzy Systems*, Prentice-Hall, New Jersey,1992.

[5] Chester M., *Neural Networks A Tutorial*, Prentice-Hall, New Jersey,1993.